MPMP Manual

vers. 1.8

monoflow
new media experience

Contents

Chapter 1

Introduction

- **MPMP** (Multi Platform Media Player) is a high performance cross platform media player for Windows, Android and iOS/OSX.
  To get the best performance on each platform it uses different media frameworks.
  - **Windows: Media Foundation or VLC**
  - **Android: Media Player**
  - **iOS/OSX: AVFoundation**
- In Unity you have only a unified C# interface you work with, so you don't have to deal with all the platform differences. **MPMP** is a media player but at the moment it is only for playing video files. Depending on the platform it can play different kind of video formats. You have to look into the documentation of the media frameworks to check if your video is encoded in the right format.
- If you have only the VLC version of MPMP the information in this document about the other versions and Media Foundation are not relevant to you.

Chapter 2

System Requirements

- **Unity** 5.2.2 or higher (Best is to use 5.3+ when on OSX)
- **Windows:**
  - Windows **8+ (Media Foundation)**
  - Windows **7+ (VLC)**
  - **DX11 only!**
  - Visual C++ Redistributable Packages for Visual Studio 2013 (https://www.microsoft.com/en-us/download/details.aspx?id=40784) (MPMP provides the installers in the vcredist folder)
- **Android:**
  - Android API level 15+ (Android 4.03+)
  - armeabi-v7a or x86 CPU
  - OpenGLES 2.0 or 3.0 as Graphic API
- **iOS:**
  - iOS 6+
  - OpenGLES 2.0 or 3.0 as Graphic API
- **OSX:**
  - 64 bit system

Chapter 3

Installation

1. Import the MPMP package from the Assetstore. You should now have a folder named MPMP with the following structure in your Unity project:

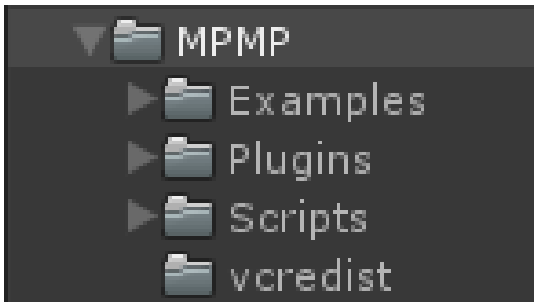


Figure 3.1: package structure

   - Examples: MPMP comes with a demo scene to show all components in a ready setup.
   - Plugins: All the native dlls and shared libraries
   - Scripts: All the C# code
   - vcredist: Installers from Microsoft (Visual C++ Redistributable Packages for Visual Studio 2013)

2. When updating from MPMP-VLC to the full MPMP version or updating an existing MPMP installation you should delete the old MPMP folder before importing. If you had entered the play mode from Unity once in your editor session you have to close Unity and restart it. Otherwise the native dlls will be cached from Unity and could not be updated properly!

3. From MPMP version 1.7 the default backend on Windows is VLC! If you want the Media Foundation backend (Windows 8+) you have to go under 'Edit/Preferences.../MPMP' and deselect VLC.

4. After installation you should see a menu item under ‘ GameObject/Create Others/MPMP. If there is no MPMP menu item you have to reimport the unitypackage or close & reopen Unity.

5. If you working on the Windows platform you need to install the Visual C++ Redistributable Packages for Visual Studio 2013 We provide the installers from Microsoft in the vcredist folder. If you don't want to install the full Redistributable Packages you need at least following dlls located beside your exe:
msvcp120.dll, msvcr120.dll, vcamp120.dll and vccorlib120.dll

Chapter 4

Usage

4.1 Setup

1. To play a video with MPMP you have to add an instance of the MPMP player to your scene. Just select the menu item under GameObject/Create Others/MPMP.

2. When you select the MPMP instance you can manage it with the component inspector.

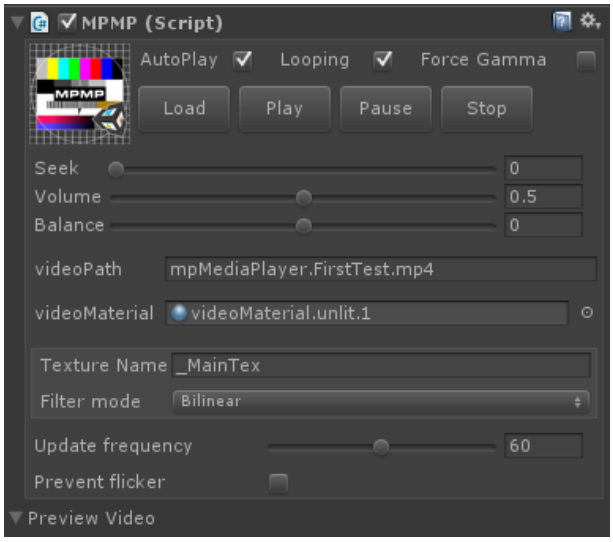


Figure 4.1: MPMP inspector

The inspector provides all the elements to interact with your media in the editor

- **Load**: Before you can play a media file/stream you have to load it into the player. MPMP loads the media file from the location you specified in the videoPath field.
- **Play**: When a media file is ready for playing(loaded) you can start playing it.
- **Pause**: If a media file is playing you can pause it. To resume you have to call **Play** again.
- **Stop**: Stops the media and seeks to the first frame.You have to call **Play** to restart the media.
- **Seek**: You can seek to a position into the media file.(doesn't work on live streams). The time you seek the media file is paused.
- **Autoplay**: The media file starts to play automatically after it is loaded.
- **Looping**: When the media file reaches the end position it jumps to the start and plays again.

- **Force Gamma**: There could be problems on Windows 10 with NVIDIA GPU using linear colorspace. This option enforces that always a rgb instead of a srgb native texture is used. But you need a gamma correction shader for the right display of your videotexture.(MPMP provides some example shaders)

3. **videoPath:** the path/url to your video.
   You can play local files, remote files(progressive streaming) and streaming video (Media Foundation doesn't seems to support streaming right out of the box). You can change the content of MPMP at every time. Just change the videoPath and trigger a new loading. For local files you can specifiy an absolute path with the `file://` sheme or without a sheme (the StreamingAssets folder is the root) For remote files or streams you just use `http://`∗yourURL∗ . Depending on you platform you can use progressiv streaming.(Your video file must be exported with the faststart option.) Examples:

   - myVideo.mp4 => StreamingAssetFolder/myVideo.mp4
   - file://C:/Folder1/myVideo.mp4 => C:/Folder1/myVideo.mp4
   - C:/Folder1/myVideo.mp4 => C:/Folder1/myVideo.mp4
   - http://www.myURL.com/Folder1/myVideo.mp4

   On Android you should use no sheme for local files. Files in the StreamingAssets folder are resolved internally so you can use just the filename. If you want to load from an external sdcard you have to use a path like this:

   - /sdcard/mydirectory/myVideo.mp4

   (And set 'Write Access' to 'External (SDCard) in your player settings!)

4. **videoMaterial:**
   Here you specify a material that should display the texture of the video (the shader needs a '_MainTex' property!). At runtime the video is rendered into the materials mainTexture. Keep in mind that except of Android the raw video texture is flipped on the y-axis. We provide a script that flips the localScale.y of a gameobject (MPMPScaleFlipY.cs) on the affected platforms so you don't need to write your own flip script.

5. **Texture Name:**
   Name of the texture property within the materials shader, default is _MainTex to support almost every shader that comes with Unity. Change this only if you use some custom shader. For example if you have a shader that uses several video textures you can use multiple MPMP instances that have the same video material but every instance uses another texture name.

6. **Filter Mode:**(Windows only)
   You can choose between Point and Bilinear texture filtering. Bilinear filtering gives better visual result when looking from flat angles onto the video texture or when you scale your video.

7. **Update frequency:**
   You can adjust the interval how often the native plugin should update the texture.Default is 60 but in low performance situations you can lower the frequency to 30 without a huge visual impact.

8. **Prevent flicker**
   If this option is enabled a screenshot from the last video frame is made when you trigger a new load to prevent texture flickering. As this is a heavy task it could cause some delay on weak hardware. For more info see Loading behaviour

9. **Preview Video:**
   When playing a video you can watch a little preview image of the video here.

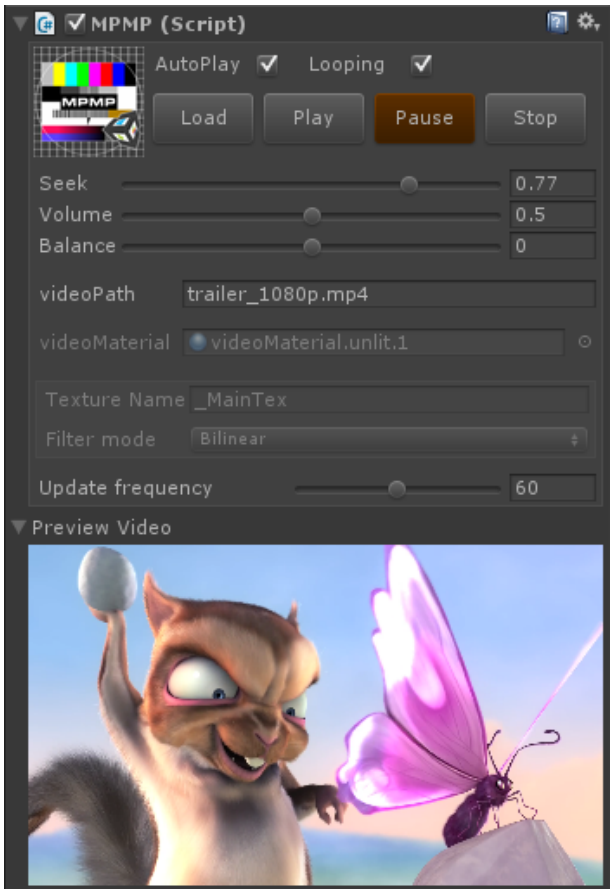


Figure 4.2: MPMP inspector

4.2 UGUI

MPMP comes with a preconfigured UGUI Prefab that you can use to control a MPMP instance at runtime.(MP↩ MP/Examples/GUI/MPMP.ugui) We provide a class MPMP_ugui_Element.cs that you can use for your own UGUI elements. (MPMP/Scripts/GUI) As every instance of this class need a reference to a MPMP player you can use the MPMP_ugui_Host.cs class in a parent gameobject as a single point to set a reference to a MPMP player. All child gameobjects that are a MPMP_ugui_Element will inherit the MPMP reference from it.
Depending on the functionality you have to add the MPMP_ugui_Element to:

- Button (LOAD,PLAY,PAUSE,STOP,AUTOPLAY)
- Slider (SEEK,VOLUME,PLAYBACKRATE)
- RawTexture (TEXTURE)
- Text (TIME)
- InputField (PATH)

4.3 Miscellaneous

- **C# API**
  All the properties of the MPMP class are accessible via c#. For a small overview how you can work with MPMP in your C# scripts we provide the MPMP_APITest.cs script. For more information you can read the Class Documentation chapter.

- **Events**
  The MPMP instance has events so you can add some callback methods. For example when the OnLoaded event is called you can check the new video size and can rescale your gameobject. For more information you can read the Class Documentation chapter.

- **Two rendering backends on Windows**
  On Windows you can choose between two native backends: Media Foundation and VLC. Every backend has it's pro and cons. If you need the best performance you should use Media Foundation. Keep in mind that you need Windows 8+. The VLC Backend has some benefits (at the cost of not the same performance as Media Foundation):

  – Windows 7 backwards compatibility
  – Youtube player
  – Better streaming support
  – Wider codec support

  To switch between the backends you just go to 'Edit/Preferences.../MPMP'.

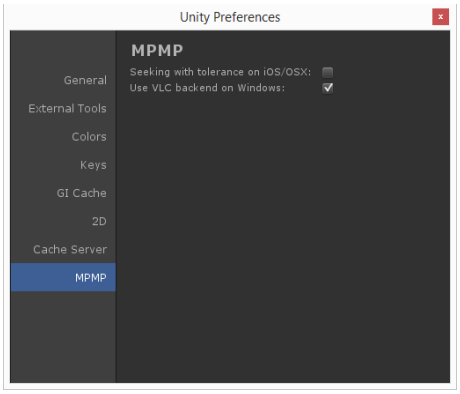


Figure 4.3: MPMP preferences

When publishing your app Unity just copies all files that are located in the plugins folder into the Data/Plugins folder of your build. MPMP has a PostProcessBuild script that does the cleanup and rearrange the files for MPMP accordingly to your selected backend.

- **Seeking**
  Seeking accuracy heavy depends on the codec and platform you use. The media framework only can seek exactly to keyframes. So you should test your video with different encoding settings to get the best results.(For example H264 is not the best codec for good seeking). When encoding your video you should make as much keyframes as possible. But this is always a difficult tradeoff between quality and filesize.
  On iOS/OSX there is a second seek mode available that specifies a tolerance value (default 0) for more precise seeking but could cause some decoding delay. To enable this seek mode you can go 'Edit/↩ Preferences.../MPMP' or you have to enable the SEEK_TOLERANCE definition at the top of the MPMP↩ _API.cs script manually.

- **Multiple audio tracks**
  If you have a project where you need localization to support several languages you can use videos that have several audio tracks incorporated. MPMP can activate an audio track at an index. The first audio track has the index 0 which is the default value if you don't specify one. Before you load your video just call **SetAudio↩ Track**(int index) . All videos that the MPMP instance plays afterwards are using the audio track at the index you have set (if there is one at this index). Changing the audio track while playing a video is not possible. You have to force a reload after changing the audio track index.

- **Loading behaviour**
  To prevent a crash when loading a video with another dimension then the current one we need to destroy the current video texture. So you will see a flicker when loading. When you are in editor mode you can add a

texture to your videomaterial that is used as default texture while loading. So you can customize the visual appearance. Otherwise you can work with the OnLoad/OnLoaded callbacks and do some scripting. The 'prevent flicker' option is another way. The last video frame is copied into the videomaterials default texture so there is no flickering when loading. This is a heavy task and could cause some delay.

- **Graphics API**
  MPMP requires on mobile platforms OpenGLES 2.0/3.0 as Graphics API. When you publish your app on iOS and forgot to adjust your player settings MPMP will automatically remove the Metal API option and write a warning to the console.

- **360° panorama viewer (Monoscopic/Stereoscopic)**
  MPMP comes with two demo scenes that has a camera setup for 360° panorama videos. You can use different layouts (side by side or top/bottom) for stereoscopic video depending on your video footage. For each eye a camera is inside a sphere that is textured with the video. To look around just rotate both cameras parent gameobject.

- **Preparing your videos for progressiv streaming**
  There are several ways to prepare your video to use progressiv streaming. The easiest is to use the Quicktime **PRO**.

  1. Open your movie with the Quicktime player **Pro**
  2. Choose File -> Export.
  3. Choose "Movie to QuickTime Movie" from the Export pop-up menu.
  4. Click Options and select video and sound compression options appropriate for web delivery.
  5. Make sure the "Prepare for Internet Streaming" checkbox is selected and Fast Start appears in the pop-up menu.

  If you have already mp4 H.264 videos and don't want to re-encode them you could use the program **MP4 FastStart** to adjust the MP4's metadata.

- **Video synchonizer**
  We provide a script that acts as a synchronizer to play several videos in a synchron way. This is for example useful if you want to display a video and do alpha masking. For such a setup you need two videos that play synchron. You specify a master that acts as a clock and your clients that synchronize their seek position to the master when the current position has a higher difference then a given threshold value.

- **WebGL support (experimental)**
  WebGL support is **experimental**. You should use it with care when using this in a production enviroment! The code is based on the 'Simple MovieTextures for Unity WebGL' demo from the Assetstore.(https://www.assetstore.unity3d.com/en/#!/content/38369) Not all features of MPMP are implemented and you should not destroy an MPMP instance while playing as there is an issue with creating more then 16 MPMP instances in one browser session. When you test your WebGL build you should put the created html and javascript files on a server.(We experienced javascript problems with local testing).

Chapter 5

FAQ

- What shader can i use?
  You can use every shader that uses a texture. MPMP uses in default the '_MainTex' property but you can change in the MPMP inspector the name of the texture property you want to use.
- Can I use MPMP on Windows 7 or Vista?
  For Windows 7 support you have to use the VLC backend.
- Can i use APK Expansion Files on Android?
  Yes. MPMP checks if the Unity app is an obb file and loads the data in a different way. You can leave your videos in the StreamingAssets folder of Unity before you split your APK at built time.
- Can i use 4K movies?
  Yes but the performance depends on your graphics card. On older cards it is possible that there is only a CPU fallback (Windows). We tested a 4K movie on Windows 8.1 with a ATI 7850 (only CPU fallback) and with a NVIDIA GTX 970 (full GPU acceleration). The VLC backend has not the same performance as Media Foundation so you need a decent hardware for 4K.
- Why is my video playing in the editor but not on my mobile device?
  Every platform has is own restrictions in terms of video codecs or audio formats. So it is most likely a problem of your video encoding. (For example at our tests on Android we had problems with a video that uses mp3 audio compression)
- I don't want to install the full redistributable packages on the target machine. What should i do?
  If you don't want to install the full redistributable packages you need at least following dlls from the packages: msvcp120.dll, msvcr120.dll, vcamp120.dll and vccorlib120.dll
  These dlls have to be located beside the main EXE of the application you publish
- Can i use MPMP for 360 videos or VR?
  Yes. It's not a problem of MPMP but more a problem of creating the right video footage and have a mesh with the right uv. We provide a 360° demo scene with a sphere that has inverted normals and and the right uv mapping. You only have to place your camera inside the sphere to watch the video. In the Links section we provide some links to video footage you can use.
- Can i calculate the amount of data that is downloaded when playing a progressive streaming video?
  You can add a callback method to the OnBuffering event. When this event is fired you can call the GetBufferLevel() method that gives you a normalized value how much data of the video file is currently downloaded.

Chapter 6

Known Issues

- There are some platform differences in terms of how remote files could be played or how they are cached on the system. Please check the video behaviour on all platforms you use. Some features don't work exactly the same way in the editor like on mobile platforms.
- On Android there is an issue with running multiple instances of MPMP at the same time and using streaming video. The native Media Player reacts on errors when your stream is disrupted. The problem is that all instances of the Media Player receive in this case a MEDIA_ERROR_SERVER_DIED error. This forces MPMP to reload the current media so all current instances of MPMP reload their content.
- On mobile platforms you should not call mpmp.Load immediately after application start. 1-2 frames delay prevents loading issues.
- Audio panning on iOS/OSX doesn't work
- Video loading on OSX 10.11 El Capitan sometimes hangs. AVFoundation on EL Capitan has some bugs so we don't recommend it at this time. We have added a small hack so if there arise a pixelbuffer error on loading we force a reload.
- On some graphics cards there could be a crash on application quit (Windows standalone). We provide a utility script where we delay the OnApplicationQuit event with a cleanup routine to shutdown all MPMP instances before the application closes.
- When using the VLC backend Unity stalls a short amount of time while loading
- There are issues on Windows 10 with NVIDIA cards using linear colorspace. If you have problems with the display of the video texture you should use the 'force gama' option and use one of the gamma correction shaders.
- When using the VLC backend on Windows and publish for 32bit we had to remove the hardware decoding.

Chapter 7

History

Version 1.8.3 - 2016.07.04

- new MPMP.dll (Removed D3D11_CREATE_DEVICE_DEBUG flag)
- fixed issues with creating Windows builds on OSX
- fixed editor issues when target platform is Windows on OSX

Version 1.8.2 - 2016.06.30

- new MPMP_VLC.dll (Removed D3D11_CREATE_DEVICE_DEBUG flag)
  -fixes crash on Windows 7 with VLC backend when Windows 8.1 SDK is missing

Version 1.8.1 - 2016.06.23

- Added WebGL support (**experimental**)
- Added gamma correction shaders

Version 1.8 - 2016.06.19

- OpenGL ES 3.0 support on mobile
- Added OnBuffering event
- Exception handling on Android improved when loading from path that could not be resolved
- Fixed problem playing videos when Time.deltaTime is null
- Added forceGamma option
- Documentation update

Version 1.7 - 2016.05.31

- VLC backend for Windows:
  -Windows 7 support
  -Better streaming support
  -Youtube player
  -Much more supported formats

- Added 'Prevent flicker' option
- Added Stop method + event (OnStop)
- Added STOP button to ugui elements
- Added SetAudioTrack(index) method. You can now use videos with multiple audio tracks and select one that should be used. Default is 0 for the first audio track
- Added HasAudioTrack(index) method to check if an audio track at index exists.
- Updated the MPMP_VR_Setup.cs script to work on VR devices without interfering Unitys camera handling
- Fixed issue with SeekToWithTolerance method missing normalized parameter
- Re-added seek property
- Fixed some errors of status properties not having the right values
- Documentation update

Version 1.6 - 2016.04.01

- API change: SeekTo(float time,bool normalized) and SeekTo(float timeInSeconds) Old seek code has to be changed from normalized values to seconds or you have to use the seekTo method with normalized = true!
- Video pauses now when you pause the editor
- Fixed bug with IsPlaying,IsPaused & IsLoading are not updating their status
- Fixed bug when native texture size was not available at OnLoaded
- New seekTo example added to MPMP_APITest.cs
- SetVideoMaterial method update
- Fixed regression bug with OnApplicationPause
- Texture is now destroyed also on ATI cards (Windows) when loading to unify the loading behaviour and prevent rare editor crashes
- GL.IssuePluginEvent is called now on every frame (Android) to prevent update issues with streaming videos
- Added internal MediaPlayer.onVideoSizeChanged callback on Android. Triggers the TEXTURE_CHANGED event and updates the internal native size variables
- iOS/OSX native libs compiled with Xcode 7.3
- Documentation update

Version 1.5.1 - 2016.03.21

- fixed memory leak when Pixelbuffer error occurs(iOS/OSX)
- fixed memory leak with FBO in the demo version (iOS/OSX)

Version 1.5 - 2016.03.18

- On Windows with NVIDIA cards the videoMaterial now displays a texture while loading. (The texture that the material has attached when you are in editor mode)
- Improved error handling on Android when a video could not be loaded
- Changed the last direct native API calls from the uGUI to a cached version
- Removed some double update callings on OSX
- SetVideoMaterial method update
- Added a PreferenceItem for MPMP to change the scripting define symbol SEEK_TOLERANCE (seek mode on iOS/OSX)
- Fixed regression bug where events are not called (Android)
- Added a video synchronizer script
- Added an Unlit AlphaMask shader
- Documentation update

Version 1.4 - 2016.03.12

- Added an option for adjusting the refresh interval of the native texture update
- linear color space on NVIDIA cards is now supported
- Loading a new video with new dimensions don't cause a crash anymore on NVIDIA cards (Windows)
- API calls from mediaPlayer uGUI don't call the plugin directly (crash on ATI cards fix)
- Pause before load fix (Windows)
- Critical Section now with higher spincount to improve stability (Windows)
- SaveAndLoad method has now new action parameter for tracing the download progress
- Changed the initialisation of MPMP to Awake. Also added a script execution order manager that forces the MPMP.cs to be executed earlier.
- MPMP_DelayQuit.cs script for shutdown all MPMP instances before your application quits. (fixes a possible crash on quit)

- New Events:
  OnPixelbufferError (OSX/iOS) forces a reload for fixing some possible video refresh problem on OSX El Capitan.
  OnTextureChanged: called when the internal texture has changed the dimensions.(Windows,Android)
- API addition: GetVideoMaterial, SetUpdateFrequency, GetUpdateFrequency
- Documentation update

Version 1.3 - 2016.02.17

- video texture is now displayed correct in linear color space
- Texture filter mode(Point/Bilinear) option for Windows
- You can now specify the texture property name that should be used in the video texture.
  So every custom shader should now work without tweaking the MPMP code by hand.(default is '_MainTex')
- Added a time ugui element for displaying the position and duration in seconds
- Added a path ugui element for managing the video path
- Added a 360° demo scene for watching stereoscopic panorama videos
- Fixed exception bug on OSX when Windows is the target platform and you run the scene in the editor
- Added a log warning and auto switch to x86_64 when publishing for OSX
- Documentation update

Version 1.2 - 2016.02.11

- Events are now called from the main thread
- SetPlaybackRate implemented :
  You can change the playback speed of the video.(negative values for reverse playback)
  On Android you need API level 23+ (Android 6+)
- OnInit event implemented
- Fixed issue with OnError events on Windows
- SeekToWithTolerance for iOS/OSX :
  This seek mode is more accurate but could cause some decoding delay
- Fixed issue with GetDuration on Android and iOS/OSX. The values are now available at OnLoaded in the right unit (seconds).
- ScaleFlipY.cs renamed to ScaleFlip.cs. The script has now the axis mode property for selecting which axis to flip
- Added a 360° demo scene for watching panorama videos
- Documentation update

Version 1.1 - 2016.02.04

- Events implemented :
  OnLoad, OnLoaded, OnPlay, OnPause, OnError, OnDestroy, OnPlaybackCompleted
- Fixed OnApplicationPaused issue. The native media player pauses now when this event is called
- API test is now in a seperate scene
- API test improved
- Fixed issue with debug dll reference on Windows x86_64
- Fixed issue with missing zip_file.jar on Android
- Fixed issue with seeking problem when using AwesomePlayer on Android
- Documentation update

Version 1.0 - 2016.01.13

- Initial release

Chapter 8

Credits

The VLC version of **MPMP** uses **libVLC** as backend.

This library is licensed under the LGPL.

- libVLC: http://www.videolan.org/vlc/libvlc.html
- LGPL: http://www.gnu.org/licenses/lgpl-2.1.html

Chapter 9

Links

Platform specific supported media formats

- **Windows**: https://msdn.microsoft.com/en-us/library/windows/desktop/dd757927(v=vs.↩85).aspx
- **Android**: http://developer.android.com/guide/appendix/media-formats.html
- **iOS**: https://developer.apple.com/library/ios/documentation/Miscellaneous/↩Conceptual/iPhoneOSTechOverview/MediaLayer/MediaLayer.html
- **VLC**: https://wiki.videolan.org/VLC_Features_Formats/

Good overview over different formats: https://en.wikipedia.org/wiki/Comparison_of_video↩_container_formats

Example 360° video footage: http://www.360heros.com/vr/

MP4 FastStart: http://www.datagoround.com/lab/

Chapter 10

Support

If you need support or have any question/suggestions please contact us.

- Email: info@monoflow.org
- Unity Forum: http://forum.unity3d.com/threads/381894

monoflow
new media experience

Chapter 11

Namespace Documentation

11.1 monoflow Namespace Reference

Classes

- class MPMP
- class ScriptOrder

Enumerations

- enum VRVideoMode { VRVideoMode.LEFT, VRVideoMode.RIGHT, VRVideoMode.TOP, VRVideoMode.B↩ OTTOM }

  Enumeration for the VR Setup
- enum FacebookVideoMode { FacebookVideoMode.MONO, FacebookVideoMode.STEREO_LEFT, FacebookVideoMode.STEREO_RIGHT }

11.1.1 Enumeration Type Documentation

11.1.1.1 enum monoflow.FacebookVideoMode [strong]

Enumerator

MONO

STEREO_LEFT

STEREO_RIGHT

11.1.1.2 enum monoflow.VRVideoMode [strong]

Enumeration for the VR Setup

Enumerator

LEFT

RIGHT

TOP

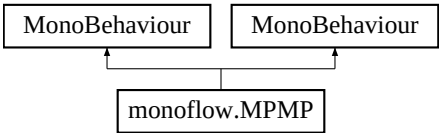
BOTTOM

Chapter 12

Class Documentation

12.1 monoflow.MPMP

Inheritance diagram for monoflow.MPMP:



Public Types

- enum Events {
  Events.LOAD, Events.LOADED, Events.PLAY, Events.PAUSE,
  Events.STOP, Events.DESTROY, Events.ERROR, Events.PLAYBACKCOMPLETED,
  Events.AVF_PIXELBUFFER_ERROR, Events.TEXTURE_CHANGED, Events.BUFFERING }

  Internal types of events from the native site
- enum FilterModeMPMP { FilterModeMPMP.Point, FilterModeMPMP.Bilinear }

  Texture filtering modes for Windows

Public Member Functions

- float GetBufferLevel ()

  Get the current percentage of the loaded data.
- void Load ()

  Triggers a loading with the current videoPath

  See also

  monoflow.MPMP.videoPath
- void Load (string path)

  Sets the current videoPath and triggers a loading
- void Play ()

  Start playing the media when loaded
- void Pause ()

  Pause the media
- void Stop ()

  Stop the media.

- float GetSeek (bool normalized)

  Retrieve the current seek position
  The values are normalized (0 - 1)
  See also

  monoflow.MPMP.GetCurrentPosition(bool normalized)
- void SeekTo (float t)

  Seek to a point in time in sec
  On iOS/OSX you can change the seek behaviour to a more precise version.
- void SeekTo (float t, bool normalized)

  Seek to a point in time
  Depending on the normalized paramter the values are normalized (0 - 1) or in sec
  On iOS/OSX you can change the seek behaviour to a more precise version.
- void SetSeeking (bool status)

  The seeking property should be set when seeking with a gui element on Android otherwise the video will not update while you seek
- void _UpdatePlaybackRate ()
- double GetCurrentPosition ()

  Get the current position of the media that is playing in sec
- double GetCurrentPosition (bool normalized)

  Get the current position of the media that is playing
  depending on the normalized parameter the values are normalized (0 -1) or in sec
- double GetDuration ()

  Get the duration of the media file in seconds
- bool IsPlaying ()

  Check if the media is playing
- bool IsPaused ()

  Check if the media is paused
- bool IsStopped ()
- bool IsLoading ()

  Check if the media is loading
- Texture2D GetVideoTexture ()

  Get the raw video texture as Texture2D
- Vector2 GetNativeVideoSize ()

  Get the video size from the native plugin as Vector2
  x: width
  y: height
- void SetVideoMaterial (Material mat, bool initFlag=false)

  Set the video material.
- Material GetVideoMaterial ()

  Get the video material.
- void SetUpdateFrequency (float interval)

  Set the frequency how often per second the native plugin should update
- float GetUpdateFrequency ()

  Get the frequency how often per second the native plugin should update
- void SetAudioTrack (int index)

  Set the index of the current selected audio track changes will be only relevant before video is loaded
- bool HasAudioTrack (int index)

  Check if an audio track at a given index is in the media that is loaded
- bool HasHadPixelBufferError ()
- IEnumerator LoadData (Uri loadUri, Action< byte[]> callBackAction, Action errorCallbackAction, Action< float > progressAction)
- IEnumerator SaveData (Uri saveUri, byte[] data, Action< bool > errorCallbackAction)

- IEnumerator DownloadAndSaveData (Uri loadUri, Uri saveUri, Action< bool > callbackAction, Action< float > progressAction)

  Use this method when you want to download a media file from a remote uri and store it local .
- IEnumerator CopyStreamingAssetData (string path)

  Copies a file from the Application.streamingAssetsPath to the Application.persistentDataPath

Static Public Member Functions

- static void MirrorUVY (MeshFilter meshf)

  Mirrors the uv.y of a mesh (1- uv.y)
- static void Set_VR_UV (MeshFilter meshf, VRVideoMode vr_mode)

Public Attributes

- Action< MPMP > OnInit

  Event that is called after the native part is initialized
- Action< MPMP > OnLoad

  Event that is called when a loading is triggert
- Action< MPMP > OnLoaded

  Event that is called when the loading is finished.
- Action< MPMP > OnPause

  Event is called when MPMP is pausing
- Action< MPMP > OnPlay

  Event is called when MPMP starts to play
- Action< MPMP > OnStop

  Event is called when MPMP stops the media
- Action< MPMP > OnDestroyed

  Event is called when the MPMP instance is destroyed
- Action< MPMP > OnError

  Event is called when there arise an error on the native site
- Action< MPMP > OnPlaybackCompleted

  Event is called when the video has reached the end.
- Action< MPMP > OnPixelBufferError

  On OSX 10.11 El Capitan there is an issue with AVFoundation.
- Action< MPMP > OnTextureChanged

  Event is called when the dimension of the video texture has changed
- Action< MPMP > OnBuffering

  Event is called when video data is buffered.You can get the current buffer level when calling GetBufferLevel() in a callback of this event.
- const string MENUITEM_NEW_MPMP = "GameObject/Create Other/MPMP/MPMP"
- const string MENUITEM_NEW_MPMP_VIDEO_SYNCHRONIZER = "GameObject/Create Other/MPMP/VideoSynchronizer"
- const string MENUITEM_NEW_MPMP_VR_SETUP = "GameObject/Create Other/MPMP/VR_Setup"
- const string MENUITEM_MPMP_COPY_VLC_DATA = "File/Copy VLC data into build %#cv"
- const string DEFAULT_TEXTURE_NAME = "_MainTex"
- const string LOGO64_NAME = "mpmp-logo.64x64"

Static Public Attributes

- static Color WARNING_COLOR = new Color(1f, 0.5f, 0f)

Properties

- bool autoPlay [get, set]

  Media starts playing automatically when loaded
- bool preventFlicker [get, set]

  If this option is enabled a copy of the current video frame is made when start loading to prevent flicker.
- float seek [get, set]

  seek property is the same as the methods SeekTo(time, normalized=true) and GetSeek(true)
- float volume [get, set]

  get or set the current volume of the media
  values are normalized (0 - 1)
- float balance [get, set]

  Set or get the current audio output balance
  normalized values:
  -1 : just left channel
  0 : both channels
  1 just right channel
  (At the moment this is unsupported on OSX/iOS)
- bool looping [get, set]

  Get or set the looping of the media
- float rate [get, set]

  Get or set the current playback rate of the media
  negative values are reverse playback
  (On Android this is only supported at API level 23+ (Android 6+))
- FilterModeMPMP filtermode [get, set]

  On Windows you can choose between Point and Bilinear texture filtering

12.1.1 Member Enumeration Documentation

12.1.1.1 enum monoflow.MPMP.Events [strong]

Internal types of events from the native site

Enumerator

LOAD
LOADED
PLAY
PAUSE
STOP
DESTROY
ERROR
PLAYBACKCOMPLETED
AVF_PIXELBUFFER_ERROR
TEXTURE_CHANGED
BUFFERING

12.1.1.2 enum monoflow.MPMP.FilterModeMPMP [strong]

Texture filtering modes for Windows

Enumerator

Point
Bilinear

12.1.2 Member Function Documentation

12.1.2.1 void monoflow.MPMP._UpdatePlaybackRate ( )

12.1.2.2 IEnumerator monoflow.MPMP.CopyStreamingAssetData (string *path*)

Copies a file from the Application.streamingAssetsPath to the Application.persistentDataPath

Parameters

path	

Returns

12.1.2.3 IEnumerator monoflow.MPMP.DownloadAndSaveData (Uri *loadUri*, Uri *saveUri*, Action< bool > *callbackAction*, Action< float > *progressAction*)

Use this method when you want to download a media file from a remote uri and store it local .

Yield as long as you download and save the data

Parameters

loadUri	
saveUri	
callbackAction	

Returns

12.1.2.4 float monoflow.MPMP.GetBufferLevel ( )

Get the current percentage of the loaded data.

(normalized 0-1). Should be called when the OnBufferng event occurs.

Returns

12.1.2.5 double monoflow.MPMP.GetCurrentPosition ( )

Get the current position of the media that is playing in sec

Returns

12.1.2.6 double monoflow.MPMP.GetCurrentPosition (bool *normalized*)

Get the current position of the media that is playing

depending on the normalized parameter the values are normalized (0 -1) or in sec

Returns

12.1.2.7 double monoflow.MPMP.GetDuration ( )

Get the duration of the media file in seconds

Returns

12.1.2.8 Vector2 monoflow.MPMP.GetNativeVideoSize ( )

Get the video size from the native plugin as Vector2

x: width

y: height

Returns

12.1.2.9 float monoflow.MPMP.GetSeek (bool *normalized*)

Retrieve the current seek position

The values are normalized (0 - 1)

See also

monoflow.MPMP.GetCurrentPosition(bool normalized)

12.1.2.10 float monoflow.MPMP.GetUpdateFrequency ( )

Get the frequency how often per second the native plugin should update

Returns

12.1.2.11 Material monoflow.MPMP.GetVideoMaterial ( )

Get the video material.

Returns

12.1.2.12 Texture2D monoflow.MPMP.GetVideoTexture ( )

Get the raw video texture as Texture2D

Returns

12.1.2.13 bool monoflow.MPMP.HasAudioTrack (int *index*)

Check if an audio track at a given index is in the media that is loaded

Parameters

index	

Returns

12.1.2.14 bool monoflow.MPMP.HasHadPixelBufferError ( )

12.1.2.15 bool monoflow.MPMP.IsLoading ( )

Check if the media is loading

Returns

12.1.2.16 bool monoflow.MPMP.IsPaused ( )

Check if the media is paused

Returns

12.1.2.17 bool monoflow.MPMP.IsPlaying ( )

Check if the media is playing

Returns

12.1.2.18 bool monoflow.MPMP.IsStopped ( )

12.1.2.19 void monoflow.MPMP.Load ( )

Triggers a loading with the current videoPath

See also

monoflow.MPMP.videoPath

12.1.2.20 void monoflow.MPMP.Load (string *path*)

Sets the current videoPath and triggers a loading

Parameters

path	

12.1.2.21 IEnumerator monoflow.MPMP.LoadData (Uri *loadUri,* Action< byte[]> *callBackAction,* Action *errorCallbackAction,* Action< float > *progressAction*)

12.1.2.22 static void monoflow.MPMP.MirrorUVY (MeshFilter *meshf*) [static]

Mirrors the uv.y of a mesh (1- uv.y)

Parameters

meshf	

12.1.2.23 void monoflow.MPMP.Pause ( )

Pause the media

12.1.2.24 void monoflow.MPMP.Play ( )

Start playing the media when loaded

12.1.2.25 IEnumerator monoflow.MPMP.SaveData (Uri *saveUri,* byte[] *data,* Action< bool > *errorCallbackAction*)

12.1.2.26 void monoflow.MPMP.SeekTo (float *t*)

Seek to a point in time in sec

On iOS/OSX you can change the seek behaviour to a more precise version.

(But could cause some decoding delay)

To set SEEK_TOLERANCE script define you can go in the Unity editor to 'Edit/Preferences.../MPMP' or enable the SEEK_TOLERANCE define at the top of the MPMP_API.cs file manually param name="t">

12.1.2.27 void monoflow.MPMP.SeekTo (float *t,* bool *normalized*)

Seek to a point in time

Depending on the normalized paramter the values are normalized (0 - 1) or in sec

On iOS/OSX you can change the seek behaviour to a more precise version.

(But could cause some decoding delay)

To set SEEK_TOLERANCE script define you can go in the Unity editor to 'Edit/Preferences.../MPMP' or enable the SEEK_TOLERANCE define at the top of the MPMP_API.cs file manually

Parameters

t	time

Parameters

normalized	normalized flag

12.1.2.28 static void monoflow.MPMP.Set_VR_UV (MeshFilter *meshf,* VRVideoMode *vr_mode*) [static]

12.1.2.29 void monoflow.MPMP.SetAudioTrack (int *index*)

Set the index of the current selected audio track changes will be only relevant before video is loaded

Parameters

index	

12.1.2.30 void monoflow.MPMP.SetSeeking (bool *status*)

The seeking property should be set when seeking with a gui element on Android otherwise the video will not update while you seek

Parameters

status	

12.1.2.31 void monoflow.MPMP.SetUpdateFrequency (float *interval*)

Set the frequency how often per second the native plugin should update

Depending on your system you normaly leave this at 60

Keep in mind that the real maximal frequency depends on the framerate of your app

Parameters

interval	

12.1.2.32 void monoflow.MPMP.SetVideoMaterial (Material *mat,* bool *initFlag* = false)

Set the video material.

(The initflag is for internal usage and should not be set unless you know what you are doing.)

Parameters

mat	

12.1.2.33 void monoflow.MPMP.Stop ( )

Stop the media.

(The media is paused and seek to 0)

12.1.3 Member Data Documentation

12.1.3.1 const string monoflow.MPMP.DEFAULT_TEXTURE_NAME = "_MainTex"

12.1.3.2 const string monoflow.MPMP.LOGO64_NAME = "mpmp-logo.64x64"

12.1.3.3 const string monoflow.MPMP.MENUITEM_MPMP_COPY_VLC_DATA = "File/Copy VLC data into build %#cv"

12.1.3.4 const string monoflow.MPMP.MENUITEM_NEW_MPMP = "GameObject/Create Other/**MPMP**/**MPMP**"

12.1.3.5 const string monoflow.MPMP.MENUITEM_NEW_MPMP_VIDEO_SYNCHRONIZER = "GameObject/Create Other/**MPMP**/VideoSynchronizer"

12.1.3.6 const string monoflow.MPMP.MENUITEM_NEW_MPMP_VR_SETUP = "GameObject/Create Other/**MPMP**/VR_Setup"

12.1.3.7 Action<**MPMP**> monoflow.MPMP.OnBuffering

Event is called when video data is buffered.You can get the current buffer level when calling GetBufferLevel() in a callback of this event.

12.1.3.8 Action<**MPMP**> monoflow.MPMP.OnDestroyed

Event is called when the MPMP instance is destroyed

12.1.3.9 Action<**MPMP**> monoflow.MPMP.OnError

Event is called when there arise an error on the native site

12.1.3.10 Action<**MPMP**> monoflow.MPMP.OnInit

Event that is called after the native part is initialized

12.1.3.11 Action<**MPMP**> monoflow.MPMP.OnLoad

Event that is called when a loading is triggert

12.1.3.12 Action<**MPMP**> monoflow.MPMP.OnLoaded

Event that is called when the loading is finished.

This is the best time to do some setup with regard to the actual video data like size, duration...

12.1.3.13 Action<**MPMP**> monoflow.MPMP.OnPause

Event is called when MPMP is pausing

12.1.3.14 Action<**MPMP**> monoflow.MPMP.OnPixelBufferError

On OSX 10.11 El Capitan there is an issue with AVFoundation.

To circumvent video refreshing errors you can catch this error.You should use this event to trigger a new Load.

12.1.3.15 Action<MPMP> monoflow.MPMP.OnPlay

Event is called when MPMP starts to play

12.1.3.16 Action<MPMP> monoflow.MPMP.OnPlaybackCompleted

Event is called when the video has reached the end.

When you are in Loop mode this event is not called!

12.1.3.17 Action<MPMP> monoflow.MPMP.OnStop

Event is called when MPMP stops the media

12.1.3.18 Action<MPMP> monoflow.MPMP.OnTextureChanged

Event is called when the dimension of the video texture has changed

12.1.3.19 Color monoflow.MPMP.WARNING_COLOR = new Color(1f, 0.5f, 0f) [static]

12.1.4 Property Documentation

12.1.4.1 bool monoflow.MPMP.autoPlay [get], [set]

Media starts playing automatically when loaded

12.1.4.2 float monoflow.MPMP.balance [get], [set]

Set or get the current audio output balance

normalized values:

-1 : just left channel

0 : both channels

1 just right channel

(At the moment this is unsupported on OSX/iOS)

12.1.4.3 FilterModeMPMP monoflow.MPMP.filtermode [get], [set]

On Windows you can choose between Point and Bilinear texture filtering

If you set this property on non Windows platforms it has no effect on the video texture

12.1.4.4 bool monoflow.MPMP.looping [get], [set]

Get or set the looping of the media

12.1.4.5 bool monoflow.MPMP.preventFlicker [get], [set]

If this option is enabled a copy of the current video frame is made when start loading to prevent flicker.

This operation could impact your performance and cause a short delay.

12.1.4.6 float monoflow.MPMP.rate [get], [set]

Get or set the current playback rate of the media

negative values are reverse playback

(On Android this is only supported at API level 23+ (Android 6+))

12.1.4.7 float monoflow.MPMP.seek [get], [set]

seek property is the same as the methods SeekTo(time, normalized=true) and GetSeek(true)

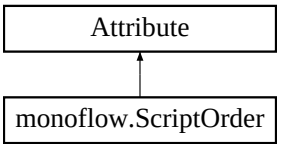
12.1.4.8 float monoflow.MPMP.volume [get], [set]

get or set the current volume of the media

values are normalized (0 - 1)

12.2 monoflow.ScriptOrder

Inheritance diagram for monoflow.ScriptOrder:



Public Member Functions

- ScriptOrder (int order)

Public Attributes

- int order

12.2.1 Constructor & Destructor Documentation

12.2.1.1 monoflow.ScriptOrder.ScriptOrder (int *order*)

12.2.2 Member Data Documentation

12.2.2.1 int monoflow.ScriptOrder.order

Index